TeCM-CLIP: Text-based Controllable Multi-attribute Face Image Manipulation

Xudong Lou[0000−0001−7311−2597], Yiguang Liu, and Xuwei Li(✉)

College of Computer Science, Sichuan University, Chengdu, Sichuan, China
featurex1994@163.com
{liuyg,lixuwei}@scu.edu.cn

Abstract. In recent years, various studies have demonstrated that utilizing the prior information of StyleGAN can effectively manipulate and generate realistic images. However, the latent code of StyleGAN is designed to control global styles, and it is arduous to precisely manipulate the property to achieve fine-grained control over synthesized images. In this work, we leverage a recently proposed Contrastive Language Image Pretraining (CLIP) model to manipulate latent code with text to control image generation. We encode image and text prompts in shared embedding space, leveraging powerful image-text representation capabilities pretrained on contrastive language images to manipulate partial style codes in the latent code. For multiple fine-grained attribute manipulations, we propose multiple attribute manipulation frameworks. Compared with previous CLIP-driven methods, our method can perform high-quality attribute editing much faster with less coupling between attributes. Extensive experimental illustrate the effectiveness of our approach. Code is available at https://github.com/lxd941213/TeCM-CLIP.

1 Introduction

Image manipulation is an interesting but challenging task, which has attracted the interest of researchers. Recent works on Generative Adversarial Networks (GANs)[9] have made impressive progress in image synthesis, which can generate photorealistic images from random latent code[14–16]. These models provide powerful generative priors for downstream tasks by acting as neural renderers. However, this synthesis process is commonly random and cannot be controlled by users. Therefore, utilizing priors for image synthesis and processing remains an exceedingly challenging task.

StyleGAN[15, 16], one of the most commonly used generative network frameworks, introduces a novel style-based generator architecture that can generate high-resolution images with unparalleled realism. Recent works[5, 30, 36] have demonstrated that StyleGAN's latent space, $\mathcal{W}$, is introduced from a learned piecewise continuous map, resulting in less entangled representations and more feasible operations. The superior properties of $\mathcal{W}$ space have attracted a host of researchers to develop advanced GAN inversion techniques[1, 2, 10, 29, 33, 34, 39] to invert real images back into the latent space of StyleGAN and perform meaningful operations. The most popular approach is to train an additional encoder to map real images to $\mathcal{W}$ space, which not only enables faithful reconstructions but also semantically meaningful edits. In addition, several methods[1–3, 29, 34, 43] recently proposed $\mathcal{W}+$ space based on $\mathcal{W}$ space, which can achieve better manipulation and reconstruction.

Existing methods for semantic control discovery include large amounts of annotated data, manual examination[10, 30, 36], or pre-trained classifiers[30]. Furthermore, subsequent operations are usually performed by moving along with directions in the latent space, using a parametric model (e.g. 3DMM in StyleRig[33]) or a trained normalized flow in StyleFlow[3]. Therefore, existing controls can only perform image operations along with preset semantic directions, which severely limits the creativity and imagination of users. Whenever additional unmapped directions are required, further manual work or large amounts of annotated data is required. Text-guided image processing[8, 19, 21–23, 28] has become feasible thanks to the development of language representations across a modality range. Recently StyleCLIP[26] achieved stunning image processing results by leveraging CLIP's powerful image-text representation. However, StyleCLIP has the following shortcomings: 1) For each text prompt, the mapper needs to be trained separately, which lacks activity in practical applications; 2) Mapping all style codes results in poor decoupling. HairCLIP[35] is designed for hair editing, it has designed a framework to quickly process hairstyles and colors for images and edit directly with textual descriptions or reference images, eliminating the need to train a separate mapper for each attribute. However, HairCLIP just edits hair attributes, it maps all the style codes like StyleCLIP, which slows down the speed and increases the coupling of attributes.

In this work, we design a model that requires fewer operations and can be disentangled more thoroughly. Overall, our model is similar to StyleCLIP[26] and HairCLIP[35]. The obvious difference is that the first two models map all style codes, while our model only needs to map partial style codes. Through extensive experiments, we found that the attributes controlled by each style code are different. If we desire to change a certain attribute of the image (such as hairstyle, emotion, etc.), we only need to perform a partially targeted style codes mapping manipulation. Our method not only exceedingly facilitates image mapping manipulation but provides less entanglement.

The contributions of this work can be summarized as follows:

1) Through extensive experiments, we analyze the properties controlled by each style code. When we need to change an attribute, we only need to manipulate the style codes that control the attribute, which can reduce the coupling between attributes while reducing the workload.

2) To further reduce the influence on other attributes, we design several decoupling structures and introduce background and face loss functions.

3) Extensive experiments and analyses reveal that our method has better manipulation quality and less training and inference time.

2 Related Work

2.1 GAN Inversion

With the rapid development of GANs, quite a few methods have been proposed to understand and control their latent space. The most normal method is GAN inversion, where the latent code most accurately reconstructs the input image from a pre-trained GAN. StyleGAN[15, 16] is used in quite a few methods due to its state-of-the-art image quality and latent spatial semantic richness. In general, there are currently three ways to embed images from the image space to the latent space: 1) learn an encoder that maps a given image to the latent space[6, 29, 34]; 2) select a random initial latent code and optimize it with gradient descent[1, 2, 18]; 3) Mix the first two methods. Between them, methods that perform optimization are better than encoders in terms of reconstruction quality but take longer. In addition, encoder-based methods can be divided into $\mathcal{W}$ and $\mathcal{W}+$ spaces after encoding. Among them, the $\mathcal{W}$ space is the latent space obtained by performing a series of fully connected layer transformations on $\mathcal{Z}$ space, which is generally considered to reflect the learning properties of entanglement better than $\mathcal{Z}$ space. $\mathcal{W}+$ space and $\mathcal{W}$ space are constructed similarly, but the latent code $w \in \mathcal{W}+$ fed to each layer of generators is different, which is frequently used for style mixing and image inversion.

2.2 Latent Space Manipulation

A host of papers[3, 29, 30, 43] propose various methods to learn semantic manipulation of latent codes and then utilize pretrained generators for image generation. Specifically, the latent space in StyleGAN[15, 16] has been manifested to enable decoupled and meaningful image manipulations. Tewari et al.[33] utilize a pre-trained 3DMM to learn semantic face edits in the latent space. Nitzan et al. [24] train an encoder to obtain a latent vector representing the identity of one image and the pose, expression, and illumination of another. Wu et al.[36] proposed to use the StyleSpace $\mathcal{S}$, and demonstrated that it is better disentangled than $\mathcal{W}$ and $\mathcal{W}+$. Collins et al.[5] perform local semantic editing by manipulating corresponding components of the latent code. These methods quintessentially follow an "invert first, manipulate later" process, first embedding the image into the latent space, and then manipulating its latent space in a semantically meaningful way. In this paper, we use a pre-trained e4e[34] model to embed images into the $\mathcal{W}+$ space, while encoding text prompt using CLIP's powerful image-text representation capabilities. Our approach is general and can be used across multiple domains without requiring domain or operation-specific data annotations.

2.3 Text-guided Image Generation and Manipulation

Reed et al.[28] generated text-guided images by training a conditional GAN conditioned on text embeddings obtained by a pre-trained encoder. Zhang et al.[40,

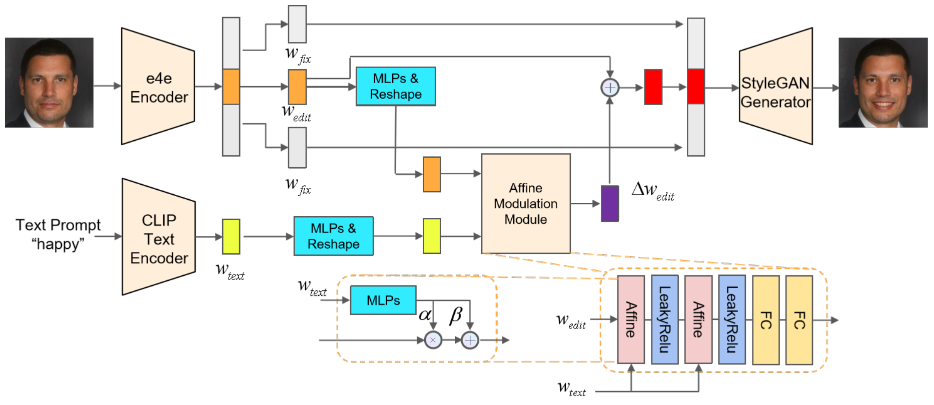

Fig. 1. We show the overall structure of our method by taking emotion ("happy") as an example. Our method supports the completion of corresponding sentiment editing according to the given text prompt, where the image is feature-mapped by pre-trained e4e[34], and the text is encoded into a 512-dimensional vector by CLIP's text encoder.

41] used multi-scale GANs to generate high-resolution images from text descriptions through a sketch-refinement process. AttnGAN[38] fused attention mechanisms between text and image features. Additional supervision is used in other works to further improve image quality. Several studies focus on text-guided image processing and kind of methods that use GAN-based encoder-decoder architectures to separate the semantics of input images and textual descriptions. ManiGAN[20] introduced a new text-image combination module that produces high-quality images. Different from the aforementioned works, StyleCLIP[26] proposed to combine the high-quality images generated by StyleGAN with the wealthy multi-domain semantics learned by CLIP[27] and use CLIP to guide the generation of images. TediGAN[37] encoded images and texts simultaneously into the latent space and then performed style mixing to generate corresponding images to realize text-guided image manipulation. Later improved versions of TediGAN[37] also used CLIP[27] for optimization operations to provide text-image similarity loss. Since StyleCLIP needs to train a separate mapper network for each text prompt, it lacks flexibility in practical applications. Therefore, HairCLIP[35] proposed a unified framework of text and image conditions to encode text or image conditions into the latent space to guide image generation.

In general, StyleCLIP[26], TediGAN[37], and HairCLIP[35] all work well for text-guided image generation, and HairCLIP outperforms the previous two models in speed. But the methods mentioned above do not decouple multiple properties well. Through extensive experiments, we discovered the properties controlled by each style code. Therefore, we merely perform feature mapping on partial style codes and achieve fewer entanglement and faster speed.

3 Proposed Method

As mentioned above, the latent space is divided into $\mathcal{Z}$ space, $\mathcal{W}$ space, and $\mathcal{W}+$ space, where $\mathcal{Z}$ space is normally distributed, and $\mathcal{W}$ space is obtained from the random noise vectors $z \in \mathcal{Z}$ via a sequence of fully connected layers. However, several studies[4, 29, 34, 36, 43] have demonstrated that inverting the images to a 512-dimensional vector $w \in \mathcal{W}$ cannot achieve accurate reconstruction. Subsequently, it has become more normal practice to encode images into an extended latent space $\mathcal{W}+$ consisting of concatenating 18 different 512-dimensional w vectors, one for each input layer of StyleGAN[15, 16]. In this paper, we perform an inversion with a pre-trained e4e[34], which maps the images to latent space, $\mathcal{W}+$, so that all style codes generated by the encoder can be recovered at the pixel level and semantic level. Since each layer of the 18-layer $\mathcal{W}+$ space controls different attributes of the image, we manipulate the corresponding style codes from $w \in \mathcal{W}+$ according to the attributes to be manipulated. At the same time, the text embedding encoded by CLIP[27] is fused into the latent space of the image. Finally, the manipulated image can be generated from the StyleGAN generator and the specific network structure is shown in Fig. 1.

3.1 Image and Text Encoders

Our approach is based on the representative of a pre-trained StyleGAN[15, 16] generator and $\mathcal{W}+$ latent space, which requires a powerful encoder to precisely encode the image into the latent domain. We select e4e[34] as our image encoder, which encodes the image into $\mathcal{W}+$ space. The latent code w is composed of 18 different style codes, which can represent the fine details of the original image more completely, accordingly, it is better than previous methods in reconstruction quality.

Unlike StyleCLIP[26], which only uses CLIP[27] as supervised loss, we refer to HairCLIP[35], which encoders the text prompts into 512-dimensional conditional embedding. Since CLIP is well trained on large-scale image-text pairs, it is easy to encode text prompts into conditional embedding and then fuse them with latent space effectively to achieve text-driven images.

3.2 Latent Mapper with Disentangled Latent Space

Quite a few works[15, 16, 31, 37] have manifested that different StyleGAN layers correspond to different semantic-level information in the generated image. StyleCLIP and HairCLIP take the same strategy, with three mappers M_c, M_m, M_f with the same network structure, which are responsible for predicting manipulated Δw corresponding to different parts of the latent code $w = (w_c, w_m, w_f)$. But in fact, this mapping method cannot disentangle the associations between various attributes well. Through extensive experiments, we found that each style code $w \in \mathcal{W}+$ controls one or more attributes, and we only need to manipulate a few of the style codes (fusion text conditional embedding and feature mapping). As an example, the first four style codes can be processed for hair manipulation.

Specifically, we first perform a four-layer full connection mapping of the style codes to be processed and also perform a four-layer full connection mapping for the text conditional embedding, and the specific formula is as follows:

$$\begin{aligned} w_{edit}^{'} &= Reshape\left(MLP\left(w_{edit}\right)\right), \\ w_{text}^{'} &= Reshape\left(MLP\left(w_{text}\right)\right). \end{aligned} \tag{1}$$

where MLP represents a four-layer fully connected layer, and $Reshape$ represents a deformation operation on the style code. $w_{edit} \in \mathbb{R}^{512}$ and $w_{text} \in \mathbb{R}^{512}$ are the style code that needs to be manipulated and the text conditional embedding, respectively. $w_{edit}^{'} \in \mathbb{R}^{4\times 512}$ and $w_{text}^{'} \in \mathbb{R}^{4\times 512}$ are the corresponding results after processing. Then processed by *Affine Modulation Module* (AMM) and then according to weights (learnable parameters) for additive fusion. We further add a two-layer fully-connected mapping for the fused style codes, which can do further information fusion. For other style codes that are not related to the attributes, we directly concatenate the original style codes and the modified style codes.

Affine Modulation Module. We design this module after referring to several methods[12, 25, 32] and making improvements. As shown in Fig. 1, after each mapper network goes through four fully connected layers, *Affine Modulation Module* (AMM) is used to fuse text-conditional embeddings to the style codes. AMM is a deep text-image fusion block that superimposes normalization layers, multiple affine transformation layers, and nonlinear activation layers (leaklyrelu) in the fusion block. AMM uses conditional embedding to modulate the style codes that were previously mapped and output in the middle. Its specific mathematical formula is as follows:

$$\begin{aligned} \alpha_1, \beta_1 &= MLP(w_{text}^{'}), \\ \alpha_2, \beta_2 &= MLP(w_{text}^{'}), \end{aligned} \tag{2}$$

$$\begin{aligned} out &= activation(\alpha_1 \times w_{edit}^{'} + \beta_1), \\ out &= activation(\alpha_2 \times out + \beta_2), \end{aligned} \tag{3}$$

$$w_{out} = MLP(out). \tag{4}$$

where out is the intermediate variable.

3.3 Loss Function

Our goal is to manipulate an attribute (e.g., hairstyle, emotion, etc.) in a disentangled way based on text prompts, while other irrelevant attributes (e.g., background, identity, etc.) need to be preserved. We introduce CLIP loss, identity loss, and latent loss used in StyleCLIP. In addition, for some attribute manipulation tasks, such as hairstyle manipulation, we design a face loss; and for the task of facial emotion manipulation, we introduce a background loss.

CLIP Loss. Our mapper is trained to operate on image attributes indicated by desired text prompts while preserving other visual attributes of the input image.

We guide the mapper by minimizing the cosine distance of the generated image and text prompts in the CLIP latent space.

$$\mathcal{L}_{\text{clip}}\left(w\right)=D_{\text{clip}}\left(G\left(Concat(w_{fix},w_{edit}+M(w_{edit},w_{text}))\right),w_{text}\right). \tag{5}$$

where M is the AMM mentioned above, G is a pre-trained StyleGAN generator and D_{clip} is the cosine distance between the CLIP embeddings of its two arguments. In addition, w_{edit} and w_{fix} are the style codes that need to be manipulated and do not need to be manipulated, respectively.

Latent Loss. To preserve the visual attributes of the original input image, we minimize the L_2 norm of the manipulation step in the latent space.

$$\mathcal{L}_{\text{latent}}(w)=\left\|M\left(w_{edit},w_{text}\right)\right\|_2. \tag{6}$$

Identity Loss. Similarity to the input image is controlled by the L_2 distance in latent space, and by the identity loss

$$\mathcal{L}_{\text{id}}(w)=1-\left\langle R\left(G\left(Concat(w_{fix},w_{edit}+M(w_{edit},w_{text}))\right)\right),R(G(w))\right\rangle. \tag{7}$$

where R is a pre-trained ArcFace[7] network for face recognition, and $\langle\cdot,\cdot\rangle$ computes the cosine similarity between it's arguments. G is the pre-trained and fixed StyleGAN generator.

Face Loss. Although only processing partial style codes could decrease attribute entanglement, the majority of style codes control multiple attributes, which leads to the issue of entanglement still not being well solved. Face loss is introduced which is mainly used to optimize and reduce the impact on the face when manipulating attributes that are not related to the face. Firstly, we use the pre-trained MTCNN[42] to detect faces and set all the values in the detected face position to 1, and all other positions to 0, and record it as the *mask*:

$$mask=MTCNN(G(w)), \tag{8}$$

$$\begin{gathered} w^{'}=Concat(w_{fix},w_{edit}+M(w_{edit},w_{text})),\\ \mathcal{L}_{\text{face}}\left(w\right)=\left\|G(w)\odot mask-G(w^{'})\odot mask\right\|_2. \end{gathered} \tag{9}$$

Background Loss. Background loss is the opposite of face loss. When we need to manipulate the face, the background loss can reduce the impact on attributes other than the face. The formula is as follows:

$$\mathcal{L}_{\text{background}}\left(w\right)=\|G(w)\odot(1-mask)-G(w^{'})\odot(1-mask)\|_2. \tag{10}$$

Our total base loss function is a weighted combination of these losses:

$$\mathcal{L}_{\text{base}}\left(w\right)=\lambda_{\text{clip}}\,\mathcal{L}_{\text{clip}}\left(w\right)+\lambda_{\text{latent}}\,\mathcal{L}_{\text{latent}}\left(w\right)+\lambda_{\text{id}}\,\mathcal{L}_{\text{id}}(w). \tag{11}$$

where λ_{clip}, λ_{latent}, λ_{id} are set to 0.6, 1.0, 0.2 respectively by default. For tasks related to face manipulation, such as emotional manipulation, age manipulation, gender manipulation, etc., we also introduce a background loss function, as follows:

$$\mathcal{L}_{\text{total}}\left(w\right)=\mathcal{L}_{\text{base}}\left(w\right)+\lambda_{\text{background}}\,\mathcal{L}_{\text{background}}\left(w\right). \tag{12}$$

And for the manipulation of hairstyle and hair color, we use face loss to reduce the impact on the face:

$$\mathcal{L}_{\text{total}}(w) = \mathcal{L}_{\text{base}}(w) + \lambda_{\text{face}}\,\mathcal{L}_{\text{face}}(w). \tag{13}$$

where both $\lambda_{\text{background}}$ and λ_{face} are set to 2.0.

Table 1. Experimental results of layer-by-layer analysis of the 18-layer StyleGAN generator. We list the attributes corresponding to all eighteen layers of style codes. Most style codes control multiple attributes. Among them, we use "micro features" to represent the style codes with low effects.

n-th	**attribute**	***n-th***	**attribute**
1	eye glasses, head pose, hair length	10	eye color, hair color
2	eye glasses, head pose, hair length	11	skin color
3	eye glasses, head pose, hair length	12	micro features
4	head pose, hair length	13	skin color, clothes color
5	hairstyle, cheekbones, mouth, eyes, nose, clothes	14	skin color, clothes color
6	mouth	15	micro features
7	eyes, gender, mustache, forehead, mouth	16	micro features
8	eyes, hair color	17	micro features
9	hair color, clothes color, lip color, mustache color, eyebrow color, eye color, skin color	18	clothes color

4 Experiments and Comparisons

To explore the efficiency and effectiveness of our method, we evaluate and compare our method with other methods on multiple image attribute manipulation tasks. The main comparison contents include hairstyle, hair color, emotion, age, gender, etc.

4.1 Implementation Details and Evaluation Metric

Implementation Details. We use e4e[34] as our inversion encoder and StyleGAN2 as our generator. We trained and evaluated our model on the CelebA-HQ dataset[13]. We collected text prompts from the internet describing attributes such as hairstyle, hair color, facial emotion, age, gender, etc. The number of text prompts for each attribute varies, ranging from 5 to 10. For each attribute mapper, it merely maps the style codes related to the attribute that needs to be manipulated and fuses the text conditional embeddings, which are randomly selected from the text set. For the training strategy, the base learning rate is 0.0001, the batch size is 1, and the number of training iterations for each attribute mapper is 20,000. We use the Adam optimizer[17], with β_1 and β_2 set to 0.9 and 0.999, respectively.

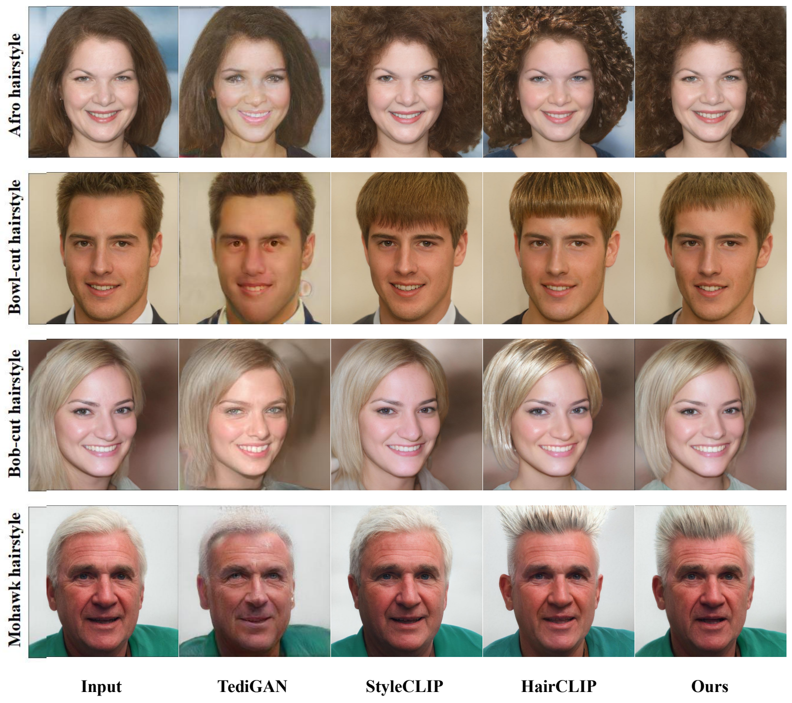

Fig. 2. Visual comparison with other methods. The left side of each row is the text prompt, the first column is the input image, and each subsequent column is the generated result of each method.

Evaluation Metric. Since our task is to edit the image, the pixel level of the image will change greatly, and we can only analyze the results according to the semantic level. Therefore, some indicators, such as PSNR, SSIM, FID[11], etc. that are commonly used in the field of image generation are not suitable for our task. We only evaluate whether the output satisfies the text prompt and whether the identity attribute is consistent with the input. For the metric of accuracy (ACC), we use CLIP[27] to calculate the similarity between text prompts and outputs, and introduce identity similarity (IDS) to evaluate the identity consistency between the outputs and the inputs.

4.2 Comparisons with State-of-the-art Methods

We mainly conduct comprehensive comparisons on the manipulation of character hairstyles, hair color, facial emotion, and other attributes. Our method merely does mapping training once for the attributes to be manipulated. Compared with mainstream methods such as StyleCLIP[26] and HairCLIP, the speed is

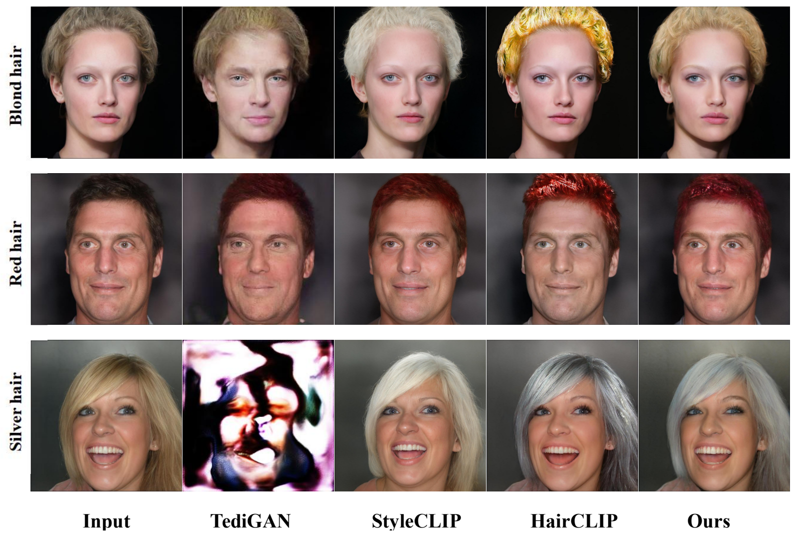

Fig. 3. Visual comparison with other methods on hair color editing.

faster and the coupling between properties is lower. As shown in Table 3, our method is less than StyleCLIP in both training and inference time. The properties controlled by each style code in the $\mathcal{W}+$ space are given in Table 1. Through extensive experimentation, and with the help of more than 20 people, which we analyzed and discussed together, we finally determined the properties controlled by each style code in the $\mathcal{W}+$ space.

Hairstyle Manipulation. We comprehensively compare our method with current state-of-the-art text-driven image processing methods TediGAN[37], StyleCLIP, and HairCLIP in hairstyle editing. According to Table 1, the *5-th* style code can control the hairstyle, while the first 4 style codes can control the length of the hair. Through extensive experiments, we found that jointly editing the first 5 layers of style codes can achieve better hairstyle editing results. Therefore, we only do feature mapping for the first 5 layers of style codes and embed text features, and do nothing for other style codes. According to the experimental results shown in Fig. 2, TediGAN fails to edit hairstyles. Both the qualitative and quantitative results of StyleCLIP are close to our method, but the issue with StyleCLIP is slow and requires training a mapper for each text prompt. And StyleCLIP does not match the text "Mohawk hairstyle" well. In general, HairCLIP has the best effect on hairstyle editing, but there is a clear difference in hair color, indicating that there is a coupling between attributes. We can find from the quantitative results in Table 2 that the results generated by HairCLIP are the closest to the text prompts but lose certain identity information compared to our method.

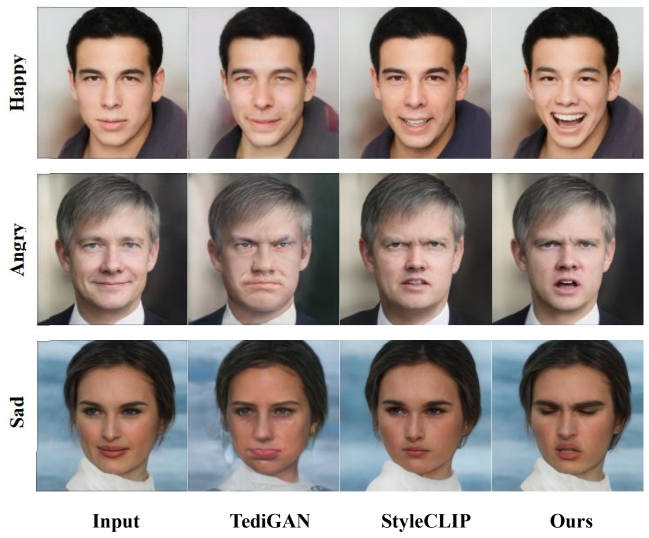

Fig. 4. Visual comparison with other methods on emotion editing.

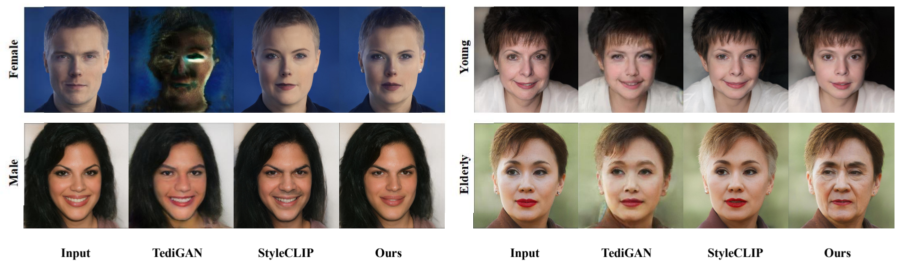

Fig. 5. Visual comparisons with other methods on gender and age editing. Gender on the left, age on the right.

Table 2. Quantitative comparison of all attribute edits, where red font indicates the best result for each column.

Method	Hairstyle		Hair color		Emotion		Gender		Age	
	ACC	IDS	ACC	IDS	ACC	IDS	ACC	IDS	ACC	IDS
TediGAN	0.1842	0.5276	0.2032	0.3647	0.2135	0.5716	0.1719	0.4816	0.1926	0.6673
StyleCLIP	0.2119	0.8574	0.2436	0.8217	0.2169	0.7808	0.2346	0.7211	0.2153	0.8037
HairCLIP	0.2367	0.8731	0.2303	0.9237						
Ours	0.2325	0.8934	0.2377	0.8732	0.2478	0.8081	0.2153	0.7331	0.2410	0.7219

Hair Color Manipulation. Same as hairstyle manipulation, we compared our method with TediGAN, StyleCLIP, and HairCLIP on hair color editing. According to Table 1, the style codes of the *8-th*, *9-th*, *10-th* layers control the color of the hair. The qualitative comparisons of experimental results are shown in Fig. 3. Intuitively, TediGAN failed on silver hair editing, and other effects were mediocre. In addition, it is easy to find that the face color of HairCLIP is different from the face color of the input. From the quantitative results in Table 2, we can see that HairCLIP preserves the identity attribute better, and StyleCLIP matches the text prompt better.
Emotion Manipulation. In addition to hair, we also experimented with facial emotion editing and made qualitative and quantitative comparisons with TediGAN and StyleCLIP. Since HairCLIP cannot operate on attributes other than hair, we omit it here. Emotions are reflected by facial features, consequently, we edit style codes that control characters' facial features. As shown in Table 1, human facial features (nose, eyes, mouth) are controlled by layer 5, layer 6, and layer 7. We use the same method to design an emotion mapper, and the generated results and the qualitative comparison results are shown in Fig. 4. The results of TediGAN are closer to the textual prompt, but it is obvious that the identity attributes vary greatly. Both StyleCLIP and our method retain perfect identity attributes, but StyleCLIP does not work well for "Sad" emotion editing. It can also be found from the quantitative analysis in Table 2 that the IDS of StyleCLIP and our method are relatively high, reaching 0.7808 and 0.8081 respectively, indicating that the identity attributes are well preserved. And the ACC of our method is also the highest among the three, reaching 0.2478, which illustrates that our method is more matched with text prompts.
Other Manipulations. In addition to the above-mentioned manipulations, we also conducted experiments and comparisons on several attributes such as age and gender. For the edits of age, our method cannot accurately generate the specified age results, and can only be edited with approximate descriptions, such as old, middle-aged, young, etc. Gender editing, it's editing women as men and men as women. These two attributes are related to human facial features, consequently, we trained these two mappers in the same way as emotion editing. The specific experimental and comparison results are shown in Fig. 5 and Table 2. Both in terms of qualitative and quantitative results, StyleCLIP and our method are close, and both are better than TediGAN. StyleCLIP achieved the best results on both the ACC indicator of gender and the IDS indicator of age, reaching 0.2346 and 0.8037, respectively. Our method achieves the best results on the IDS indicator of gender and the ACC indicator of age, reaching 0.7331 and 0.2410, respectively.

4.3 Ablation Study

To demonstrate the effectiveness of each component in our overall approach, we perform ablation studies by evaluating the following subset models and loss functions:

Table 3. Quantitative time comparison on a single 1080Ti GPU, where red fonts indicate the best results.

Method	train time	infer.time
StyleCLIP	10-12h	75ms
Ours	8h	43ms

Table 4. Quantitative comparison of face loss and background loss, where red fonts indicate the best results.

	Hairstyle			Emotion	
λ_{face}	ACC	IDS	λ_{bg}	ACC	IDS
0	0.2308	0.8636	0	0.2387	0.7988
1	0.2347	0.8742	1	0.2412	0.8010
2	0.2325	0.8934	2	0.2478	0.8081
3	0.2136	0.8898	3	0.2296	0.8103

Table 5. For the number of edited style codes, quantitative comparisons are made on emotion editing and model scale, respectively, where red fonts indicate the best results.

Method	Emotion		Mapping module	
	ACC	IDS	Params(M)	MFLOPs
Edit all style codes	0.2394	0.6083	170.2	170.1
Ours	0.2478	0.8081	28.37	28.35

Loss Function. We verify the effectiveness of face loss and background loss by controlling variables, the results are shown in Fig. 6. To verify the importance of the background loss, we did a comparative experiment on emotion editing, and the experimental results are shown in the left half of Fig. 6. It can be seen that when the background loss is not used, the generated results are significantly different from the input on the hair, and after using the background loss, the background similarity can be well maintained. Likewise, we conduct comparative experiments on hairstyle editing to verify the effectiveness of face loss. The eyes and mouth have changed to a large extent without using face loss, and after using face loss, these changes are gone. The above two sets of comparative experiments fully verify the effectiveness of our background loss and face loss. It should be pointed out that these two losses limit the editing ability of images. As shown in Table 4, face loss can reduce the loss of identity information, but it limits the editing ability of images to a certain extent. Compared with face loss, the effect of background loss is limited. When we do not use these two losses, our ACC and IDS metrics both exceed StyleCLIP, which can also reflect the effectiveness of our $\mathcal{W}+$ feature selection.

Mapping All Style Vectors. To prove the effectiveness and efficiency of using partial style codes, we map all style codes according to StyleCLIP. As can be seen from Fig. 7, when mapping all style codes, the generated results will look unnatural and cannot match the text prompts well. We also give the quantitative comparison results in Table 5. Mapping all style codes not only increased the number of parameters and FLOPs several times, but also worse the ACC and IDS indicators, even the IDS decreased by nearly 0.2, which also proves that our method not only reduces the number of parameters speeding up the training and inference time but also generate better results.

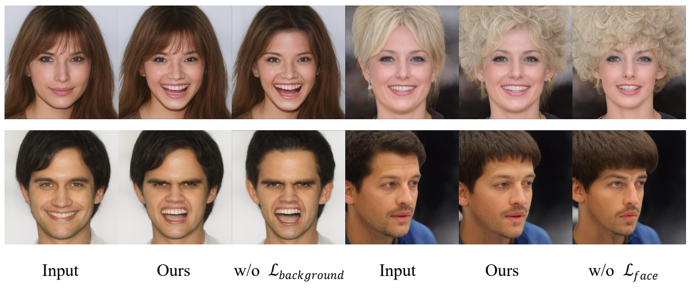

Fig. 6. Visual comparison of our generated results with methods and variants of our model. The top left and bottom left are comparisons of sentiment, two text prompts are "happy" and "angry", where the left is the original input, the middle is our result, and the right is the result without background loss. The top right and bottom right are comparisons of hairstyles, the two text prompts are "afro hairstyle" and "bowl-cut hairstyle", where the left is the original input, the middle is our result, and the right is the result without face loss.

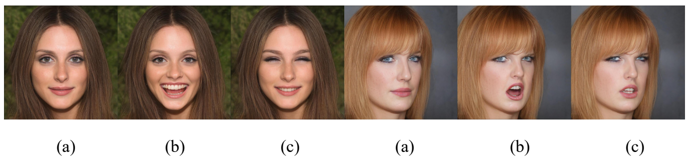

Fig. 7. Visual comparison of our generated results with methods and variants of our model. The text prompts are "happy" and "angry", respectively. (a) The input images. (b) Results were obtained with our method (map part of the style codes). (c) Results were obtained with "map all style codes".

5 Conclusions

We propose a new image attribute manipulation method, which combines the powerful generative ability of StyleGAN with the extraordinary visual concept encoding ability of CLIP, which can easily embed text prompts into images and guide image generation. Our model support high-quality manipulation of multiple attributes, including emotion editing, hairstyle editing, age editing, etc., in a decoupled manner. Extensive experiments and comparisons demonstrate that our method outperforms previous methods in terms of operational capability, irrelevant attribute preservation, and image realism.

It should be pointed out that our method sometimes fails to edit certain colors (blue, yellow, etc.) in hair color editing. We suspect that the editor's style codes are not complete enough, or that CLIP's text encoder does not work well with colors. Another point is that although our method is faster than StyleCLIP, one mapper cannot edit all attributes. In the future, we will continue to refine and improve these issues.

References

1. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan: How to embed images into the stylegan latent space? In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4432–4441 (2019)
2. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan++: How to edit the embedded images? In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8296–8305 (2020)
3. Abdal, R., Zhu, P., Mitra, N.J., Wonka, P.: Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. ACM Transactions on Graphics (ToG) **40**(3), 1–21 (2021)
4. Alaluf, Y., Patashnik, O., Cohen-Or, D.: Restyle: A residual-based stylegan encoder via iterative refinement. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6711–6720 (2021)
5. Collins, E., Bala, R., Price, B., Susstrunk, S.: Editing in style: Uncovering the local semantics of gans. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5771–5780 (2020)
6. Creswell, A., Bharath, A.A.: Inverting the generator of a generative adversarial network. IEEE transactions on neural networks and learning systems **30**(7), 1967–1974 (2018)
7. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: Arcface: Additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4690–4699 (2019)
8. Dong, H., Yu, S., Wu, C., Guo, Y.: Semantic image synthesis via adversarial learning. In: Proceedings of the IEEE international conference on computer vision. pp. 5706–5714 (2017)
9. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
10. Härkönen, E., Hertzmann, A., Lehtinen, J., Paris, S.: Ganspace: Discovering interpretable gan controls. Advances in Neural Information Processing Systems **33**, 9841–9850 (2020)
11. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017)
12. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: Proceedings of the IEEE international conference on computer vision. pp. 1501–1510 (2017)
13. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196 (2017)
14. Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J., Aila, T.: Alias-free generative adversarial networks. Advances in Neural Information Processing Systems **34**, 852–863 (2021)
15. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4401–4410 (2019)
16. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8110–8119 (2020)

17. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
18. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
19. Koh, J.Y., Baldridge, J., Lee, H., Yang, Y.: Text-to-image generation grounded by fine-grained user attention. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 237–246 (2021)
20. Li, B., Qi, X., Lukasiewicz, T., Torr, P.H.: Manigan: Text-guided image manipulation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7880–7889 (2020)
21. Li, W., Zhang, P., Zhang, L., Huang, Q., He, X., Lyu, S., Gao, J.: Object-driven text-to-image synthesis via adversarial training. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12174–12182 (2019)
22. Liu, Y., De Nadai, M., Cai, D., Li, H., Alameda-Pineda, X., Sebe, N., Lepri, B.: Describe what to change: A text-guided unsupervised image-to-image translation approach. In: Proceedings of the 28th ACM International Conference on Multimedia. pp. 1357–1365 (2020)
23. Nam, S., Kim, Y., Kim, S.J.: Text-adaptive generative adversarial networks: manipulating images with natural language. Advances in neural information processing systems **31** (2018)
24. Nitzan, Y., Bermano, A., Li, Y., Cohen-Or, D.: Face identity disentanglement via latent space mapping. arXiv preprint arXiv:2005.07728 (2020)
25. Park, T., Liu, M.Y., Wang, T.C., Zhu, J.Y.: Semantic image synthesis with spatially-adaptive normalization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2019)
26. Patashnik, O., Wu, Z., Shechtman, E., Cohen-Or, D., Lischinski, D.: Styleclip: Text-driven manipulation of stylegan imagery. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2085–2094 (2021)
27. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning. pp. 8748–8763. PMLR (2021)
28. Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H.: Generative adversarial text to image synthesis. In: International conference on machine learning. pp. 1060–1069. PMLR (2016)
29. Richardson, E., Alaluf, Y., Patashnik, O., Nitzan, Y., Azar, Y., Shapiro, S., Cohen-Or, D.: Encoding in style: a stylegan encoder for image-to-image translation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2287–2296 (2021)
30. Shen, Y., Gu, J., Tang, X., Zhou, B.: Interpreting the latent space of gans for semantic face editing. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9243–9252 (2020)
31. Shi, Y., Yang, X., Wan, Y., Shen, X.: Semanticstylegan: Learning compositional generative priors for controllable image synthesis and editing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11254–11264 (2022)
32. Tao, M., Tang, H., Wu, F., Jing, X.Y., Bao, B.K., Xu, C.: Df-gan: A simple and effective baseline for text-to-image synthesis. arXiv e-prints (2020)
33. Tewari, A., Elgharib, M., Bharaj, G., Bernard, F., Seidel, H.P., Pérez, P., Zollhofer, M., Theobalt, C.: Stylerig: Rigging stylegan for 3d control over portrait images.

In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6142–6151 (2020)
34. Tov, O., Alaluf, Y., Nitzan, Y., Patashnik, O., Cohen-Or, D.: Designing an encoder for stylegan image manipulation. ACM Transactions on Graphics (TOG) **40**(4), 1–14 (2021)
35. Wei, T., Chen, D., Zhou, W., Liao, J., Tan, Z., Yuan, L., Zhang, W., Yu, N.: Hairclip: Design your hair by text and reference image. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18072–18081 (2022)
36. Wu, Z., Lischinski, D., Shechtman, E.: Stylespace analysis: Disentangled controls for stylegan image generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12863–12872 (2021)
37. Xia, W., Yang, Y., Xue, J.H., Wu, B.: Tedigan: Text-guided diverse face image generation and manipulation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2256–2265 (2021)
38. Xu, T., Zhang, P., Huang, Q., Zhang, H., Gan, Z., Huang, X., He, X.: Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1316–1324 (2018)
39. Yang, C., Shen, Y., Zhou, B.: Semantic hierarchy emerges in deep generative representations for scene synthesis. International Journal of Computer Vision **129**(5), 1451–1466 (2021)
40. Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., Metaxas, D.N.: Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In: Proceedings of the IEEE international conference on computer vision. pp. 5907–5915 (2017)
41. Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., Metaxas, D.N.: Stackgan++: Realistic image synthesis with stacked generative adversarial networks. IEEE transactions on pattern analysis and machine intelligence **41**(8), 1947–1962 (2018)
42. Zhang, K., Zhang, Z., Li, Z., Qiao, Y.: Joint face detection and alignment using multitask cascaded convolutional networks. IEEE signal processing letters **23**(10), 1499–1503 (2016)
43. Zhu, J., Shen, Y., Zhao, D., Zhou, B.: In-domain gan inversion for real image editing. In: European conference on computer vision. pp. 592–608. Springer (2020)